# Converting untyped formulas to typed ones 

Jan Van den Bussche Luca Cabibbo


#### Abstract

We observe that every first-order logic formula over the untyped version of some many-sorted vocabulary is equivalent to a union of manysorted formulas over that vocabulary. This result has as direct corollary a theorem by Hull and Su on the expressive power of active-domain quantification in the relational calculus.


Keywords: Many-sorted logic, relational database, relational calculus

## 1 Introduction

Many-sorted logic is widely used in the formal aspects of computer science as a "typed" version of classical (first-order) logic which is in principle "untyped", or, as one also calls it, one-sorted. In this short paper, we compare many-sorted and one-sorted first-order logic with respect to their expressive power. It is well-known that many-sorted logic can be simulated using one-sorted logic. We prove a converse to this simulation.

Specifically, suppose we work in some fixed many-sorted vocabulary $\sigma$. There are two basic ways to express first-order logic properties of $\sigma$-structures. One can use standard many-sorted $\sigma$-formulas, which are well-typed in the sense that each variable occurring in them is restricted to range only over elements of a specified sort. Or one can use untyped formulas, where variables can range over elements of all sorts, and in which predicates are available to test whether an element belongs to a particular sort. It is well known that every well-typed formula is equivalent to an untyped formula. We show that conversely, every untyped formula is equivalent to a union of well-typed formulas. (The union appears simply because a single untyped formula can be true for differentlysorted valuations of its free variables, which is impossible for well-typed formulas by their very nature.)

As an application, we show how this observation yields as a direct corollary an important theorem by Hull and Su on the expressive power of active-domain quantification in the relational calculus as a query language for relational databases. We thus obtain a simple proof of this theorem, which was originally proven in a rather complicated way.

The reader is supposed to be familiar with elementary mathematical logic.

## 2 Definitions

We start by defining vocabularies and structures over a vocabulary.
Definition 1 A (relational) vocabulary is a triple $\sigma=\left(\sigma_{1}, \sigma_{2}, \sigma_{3}\right)$, where

1. $\sigma_{1}$ is a finite set of symbols called sort symbols;
2. $\sigma_{2}$ is a set of symbols called relation symbols; and
3. $\sigma_{3}$ is a mapping on the set of relation symbols, assigning to each relation symbol $R$ a tuple $\sigma_{3}(R)$ of sort symbols.

The tuple $\sigma_{3}(R)$ is called the type of $R$ in $\sigma$. Generally, a tuple of sort symbols is called a type.

Without ambiguity we will drop the subscript 3 and henceforth write $\sigma(R)$ instead of $\sigma_{3}(R)$.

Definition 2 A structure over a vocabulary $\sigma$ is a pair $A=\left(A_{1}, A_{2}\right)$, where

1. $A_{1}$ is a function mapping each sort symbol $s$ of $\sigma$ to a set $A_{1}(s)$, such that different sort names get disjoint sets; and
2. $A_{2}$ is a function mapping each relation symbol $R$ of $\sigma$ to a subset $A_{3}(R)$ of $A_{1}\left(s_{1}\right) \times \cdots \times A_{1}\left(s_{n}\right)$, if $\sigma(R)=\left(s_{1}, \ldots, s_{n}\right)$.

Generally, any subset of $A_{1}\left(s_{1}\right) \times \cdots \times A_{1}\left(s_{n}\right)$ is called a relation of type $\left(s_{1}, \ldots, s_{n}\right)$ on $A$ and its elements are called tuples of type $\left(s_{1}, \ldots, s_{n}\right)$ on $A$. The elements of the set $A_{1}(s)$, for a sort symbol $s$, are called the elements of sort s in A.

Without ambiguity we will drop the subscripts 1 and 2 and henceforth write $A(s)$ instead of $A_{1}(s)$ and $A(R)$ instead of $A_{2}(R)$.

If a vocabulary contains only one sort symbol then we call it one-sorted. We can associate a one-sorted vocabulary to each vocabulary as follows:

Definition 3 Let $\sigma$ be a vocabulary. The untyped vocabulary associated with $\sigma$, denoted by $\bar{\sigma}$, is the vocabulary defined as follows:

1. There is only one sort symbol, 1 ;
2. The relation symbols of $\bar{\sigma}$ are the relation symbols of $\sigma$ plus the sort symbols of $\sigma$;
3. For each sort symbol $s$ of $\sigma, \bar{\sigma}(s):=(1)$, and for each relation symbol $R$ of $\sigma, \bar{\sigma}(R):=(1, \ldots, 1), n$ times, where $n$ is the length of the tuple $\sigma(R)$.

A structure over a one-sorted vocabulary is also called one-sorted. We can view every structure as a one-sorted structure in the following way:

Definition 4 Let $A$ be a structure over $\sigma$. The untyped structure associated with $A$, denoted by $\bar{A}$, is the structure over $\bar{\sigma}$ defined as follows. We have $\bar{A}(1):=$ $\bigcup_{s} A(s)$, where the union ranges over all sort symbols $s$ of $\sigma$. Furthermore, we have $\bar{A}(s):=A(s)$ for each sort symbol $s$ of $\sigma$ and $\bar{A}(R):=A(R)$ for each relation symbol $R$ of $\sigma$.

Now fix a vocabulary $\sigma$. For each sort symbol $s$ of $\sigma$, we assume given an unbounded supply of variables of sort s. A formula over $\sigma$ is a standard firstorder logic formula $\varphi$ in the language of equality and the relation symbols of $\sigma$, such that all variables occuring in $\varphi$ are of a sort of $\sigma$.

Let $A$ be a structure over $\sigma$ and let $\varphi$ be a formula over $\sigma$ with free variables $x_{1}, \ldots, x_{n}$. Let the sort of $x_{i}$ be $s_{i}$, and let $a_{i} \in A\left(s_{i}\right)$ for each $i=1, \ldots, n$. By $A \vDash \varphi\left[a_{1}, \ldots, a_{n}\right]$, we denote that $\varphi$ is true in $A$ under the substitution of $a_{i}$ for $x_{i}$ for $i=1, \ldots, n$. The notion of truth is the standard one, with the important provision that for each quantified variable $x$ in $\varphi$, if the sort of $x$ is $s$ then $x$ ranges only over the elements of sort $s$ in $A$. The relation defined by $\varphi$ on $A$, denoted by $A(\varphi)$, is then defined as

$$
\left\{\left(a_{1}, \ldots, a_{n}\right) \mid a_{i} \in A\left(s_{i}\right) \text { for } i=1, \ldots, n \text { and } A \mid=\varphi\left[a_{1}, \ldots, a_{n}\right]\right\}
$$

Note that this is a relation of type $\left(s_{1}, \ldots, s_{n}\right)$ on $A$.

## 3 From untyped to well-typed

We are now ready to prove the following basic lemma. The reason why it holds is the condition on structures imposed in Definition 2 that sets of elements of different sorts must be disjoint.

Lemma 1 Let $\sigma$ be a vocabulary. For every formula $\psi$ over the associated untyped vocabulary $\bar{\sigma}$ there exist a finite number of formulas $\varphi_{1}, \ldots, \varphi_{k}$ over $\sigma$ such that for each structure $A$ over $\sigma$,

$$
A(\psi)=A\left(\varphi_{1}\right) \cup \cdots \cup A\left(\varphi_{k}\right)
$$

The above equation will also be written in short as $\psi \equiv \varphi_{1} \cup \cdots \cup \varphi_{k}$.
Proof. By induction on $\psi$.

- $\psi$ is $x=y$. Then $k$ is the number of sort symbols of $\sigma$, and $\varphi_{i}$ is $x_{i}=y_{i}$, where $x_{i}$ and $y_{i}$ are variables of the $i$ th sort of $\sigma$, for $i=1, \ldots, k$.
- $\psi$ is $s(x)$, with $s$ a sort symbol. Then $k=1$ and $\varphi_{1}$ is $y=y$ with $y$ a variable of sort $s$.
- $\psi$ is $R\left(x_{1}, \ldots, x_{n}\right)$, with $R$ a relation symbol. Then $k=1$ and $\varphi_{1}$ is $R\left(y_{1}, \ldots, y_{n}\right)$, where $y_{i}$ is a variable of sort $s_{i}$, for $i=1, \ldots, k$, if $\sigma(R)=$ $\left(s_{1}, \ldots, s_{n}\right)$.
- $\psi$ is $(\chi \vee \theta)$. By induction, we have $\chi \equiv \varphi_{1} \cup \cdots \cup \varphi_{l}$ and $\theta \equiv \vartheta_{1} \cup \cdots \cup \vartheta_{m}$. Let the number of variables occurring free in $\chi$ but not in $\theta$ be $k$, and let the number of variables occurring free in $\theta$ but not in $\chi$ be $n$. Then

$$
\psi \equiv \bigcup_{\vec{t}}\left(\varphi_{1} \wedge \phi_{\vec{t}}\right) \cup \cdots \cup \bigcup_{\vec{t}}\left(\varphi_{l} \wedge \phi_{\vec{t}}\right) \cup \bigcup_{\overrightarrow{t^{\prime}}}\left(\vartheta_{1} \wedge \phi_{\overrightarrow{t^{\prime}}}\right) \cup \cdots \cup \bigcup_{\vec{t}^{\prime}}\left(\vartheta_{m} \wedge \phi_{\overrightarrow{t^{\prime}}}\right),
$$

where $\vec{t}$ ranges over all sorts of length $n, \overrightarrow{t^{\prime}}$ ranges over all sorts of length $k$, and $\phi_{\vec{t}}$, for a sort $\vec{t}$, is the trivial formula defining the full relation of sort $\vec{t}^{1}{ }^{1}$ The variables used in the formulas $\phi_{\vec{t}}$ and $\phi_{\vec{t}^{\prime}}$ must not occur in the formulas $\varphi_{i}$ and $\vartheta_{j}$.

- $\psi$ is $(\exists x) \chi$. By induction, we have $\chi \equiv \varphi_{1} \cup \cdots \cup \varphi_{k}$. We distinguish two possibilities:
$-x$ does not occur free in $\chi$. Then $\psi$ is equivalent to $\chi$ and thus $\psi \equiv \varphi_{1} \cup \cdots \cup \varphi_{k}$.
$-x$ occurs free in $\chi$. If $\psi$ has $n$ free variables, then $\chi$ has $n+1$, and we may assume that $x$ is the $n+1$ th. Now let $y_{i}$ be the $n+1$ th free variable of $\varphi_{i}$, for $i=1, \ldots, k$. Then $\psi \equiv\left(\exists y_{1}\right) \varphi_{1} \cup \cdots \cup\left(\exists y_{k}\right) \varphi_{k}$.
- $\psi$ is $(\neg \chi)$. This is the only case that is not entirely trivial. By induction, we have $\chi \equiv \varphi_{1} \cup \cdots \cup \varphi_{l}$. For every structure $A, \bar{A}(\psi)$ is the complement of $\bar{A}(\chi)$. Assume $\psi$ has $n$ free variables. Let $\overrightarrow{s_{1}}, \ldots, \overrightarrow{s_{k}}$ be an enumeration of all sorts of length $n$ over $\sigma$. We may assume that $\varphi_{i}$ defines a relation of sort $\overrightarrow{s_{i}}$, for $i=1, \ldots, l$ (note that $l \leq k$ ). Since the $n$-tuples on $A$ can be partitioned according to their sort, the complement of $\bar{A}(\chi)$ consists of the union over $i=1, \ldots, k$ of the complements of $A\left(\varphi_{i}\right)$ taken within the class of tuples of sort $\overrightarrow{s_{i}}$, together with all tuples of some sort $\vec{s}_{j}$ with $j>l$. Hence, we have $\psi \equiv\left(\neg \varphi_{1}\right) \cup \cdots \cup\left(\neg \varphi_{l}\right) \cup \varphi_{l+1} \cup \cdots \cup \varphi_{k}$, where for $j>l$ the formula $\varphi_{j}$ is the trivial formula defining the full relation of sort $\overrightarrow{s_{j}}$.

This completes the proof of the lemma.
As an immediate corollary, we obtain:
Proposition 1 Let $\sigma$ be a vocabulary, and let $\psi$ be a formula over the associated untyped vocabulary $\bar{\sigma}$. Assume $\psi$ has $n$ free variables, and let $\left(s_{1}, \ldots, s_{n}\right)$ be a type over $\sigma$. Then there exists a formula $\varphi$ over $\sigma$ such that for each structure $A$ over $\sigma$, the relation

$$
\begin{equation*}
\left\{\left(a_{1}, \ldots, a_{n}\right) \mid a_{i} \in A\left(s_{i}\right) \text { for } i=1, \ldots, n \text { and } \bar{A} \models \psi\left[a_{1}, \ldots, a_{n}\right]\right\} \tag{*}
\end{equation*}
$$

equals $A(\varphi)$.

[^0]Proof. Let $x_{1}, \ldots, x_{n}$ be the free variables of $\psi$, and consider the formula

$$
\chi:=\psi \wedge s_{1}\left(x_{1}\right) \wedge \cdots \wedge s_{n}\left(x_{n}\right)
$$

Then $\bar{A}(\chi)$ is precisely the relation (*). Applying the lemma to $\chi$ yields formulas $\varphi_{1}, \ldots, \varphi_{k}$ over $\sigma$ such that $\bar{A}(\chi)=A\left(\varphi_{1}\right) \cup \cdots \cup A\left(\varphi_{k}\right)$. Since $\bar{A}(\chi)$ is a relation of type $\left(s_{1}, \ldots, s_{n}\right)$, the same is true for each $A\left(\varphi_{i}\right)$ and hence we can put $\varphi:=\varphi_{1} \vee \cdots \vee \varphi_{k}$, where we identify the free variables of the different $\varphi_{i}$. Thus the proposition holds.

Note that by specializing the proposition to sentences (formulas without free variables), we get that for each one-sorted sentence $\psi$ there exists a many-sorted sentence $\varphi$ such that $\bar{A}=\psi$ iff $A=\varphi$ for each $A$.

Example 1 As an illustration, assume $\sigma$ has as sort symbols 1, 2 and 3, and as relation symbols $R$ and $S$ with $\sigma(R)=(1,2)$ and $\sigma(S)=(1,3)$. Consider the untyped formula $\psi(x)=(\exists y)(R(x, y) \vee S(x, y))$. This formula defines a relation of sort (1). To find a well-typed formula equivalent to $\psi$ we first proceed according to the lemma. We have $R(x, y) \equiv R\left(x_{1}, x_{2}\right)$, where $x_{1}$ is of sort 1 and $x_{2}$ of sort 2. Similarly, $S(x, y) \equiv S\left(z_{1}, z_{3}\right)$, where $z_{1}$ is of sort 1 and $z_{3}$ of sort 3 . We thus have $\psi \equiv\left(\exists x_{2}\right) R\left(x_{1}, x_{2}\right) \cup\left(\exists z_{3}\right) S\left(z_{1}, z_{3}\right)$. We can now identify the free variables $x_{1}$ and $z_{1}$ as in the proof of the proposition and obtain the well-typed formula $\varphi(x)=\left(\exists x_{2}\right) R\left(x, x_{2}\right) \vee\left(\exists z_{3}\right) S\left(x, z_{3}\right)$ equivalent to $\psi$.

Now consider the untyped sentence $\chi=(\forall x)(1(x) \vee 2(x))$. Then for any structure $A$ over $\sigma, \bar{A}=\chi$ iff $3^{A}=\emptyset$ (i.e., there are no elements of sort 3 in $A)$. To find a many-sorted sentence equivalent to $\chi$ we again proceed first as in the lemma. We have $1(x) \equiv\left(x_{1}=x_{1}\right)$ and $2(x) \equiv\left(x_{2}=x_{2}\right)$, with $x_{1}$ of sort 1 and $x_{2}$ of sort 2. Rewriting $\chi$ as $\neg(\exists x) \neg(1(x) \vee 2(x))$, we further have $\neg(1(x) \vee 2(x)) \equiv \neg\left(x_{1}=x_{1}\right) \cup \neg\left(x_{2}=x_{2}\right) \cup\left(x_{3}=x_{3}\right)$, with $x_{3}$ of sort 3, and thus $(\exists x) \neg(1(x) \vee 2(x)) \equiv\left(\exists x_{1}\right) \neg\left(x_{1}=x_{2}\right) \cup\left(\exists x_{2}\right) \neg\left(x_{2}=x_{2}\right) \cup\left(\exists x_{3}\right)\left(x_{3}=x_{3}\right)$. Since there are no longer free variables, we can replace each $\cup$ by $\vee$ and apply the final negation to the disjunction thus obtained:

$$
\chi \equiv \neg\left(\left(\exists x_{1}\right) \neg\left(x_{1}=x_{1}\right) \vee\left(\exists x_{2}\right) \neg\left(x_{2}=x_{2}\right) \vee\left(\exists x_{3}\right)\left(x_{3}=x_{3}\right)\right)
$$

or equivalently

$$
\chi \equiv\left(\forall x_{1}\right)\left(x_{1}=x_{1}\right) \wedge\left(\forall x_{2}\right)\left(x_{2}=x_{2}\right) \wedge \neg\left(\exists x_{3}\right)\left(x_{3}=x_{3}\right) .
$$

The first two factors in the above conjunction are trivial and the third one indeed expresses that there are no elements of sort 3 .

## 4 Active-domain quantification

Let $L$ be a finite relational first-order language, i.e., a finite set of relation symbols where each relation symbol has an associated arity. Let $U$ be some fixed infinite universe of data elements. A (relational) database over $L$ is an $L$-structure, in the standard sense of mathematical logic, the domain of which equals $U$ and all of whose relations are finite. The active domain of a database $D$ is the (finite) set of domain elements that actually appear in one of the relations of $D$.

Let $\psi\left(x_{1}, \ldots, x_{n}\right)$ be a first-order formula (with equality) over $L$. If $d_{1}, \ldots, d_{n}$ are elements of $U$ we denote the truth of $\psi$ on $D$ with $d_{i}$ substituted for $x_{i}$, for $i=1, \ldots, n$, by $D \vDash \psi\left[d_{1}, \ldots, d_{n}\right]$ as usual. With the notation $D \neq_{\text {adom }}$ $\psi\left[d_{1}, \ldots, d_{n}\right]$, we mean that $\psi\left[d_{1}, \ldots, d_{n}\right]$ evaluates to true on $D$ when we let the quantifiers in $\psi$ range over the active domain of $D$ only, rather than over the whole of $U$.

Hull and $\mathrm{Su}^{2}$ showed that active-domain quantification is equally powerful as normal quantification:

Theorem 1 For each $\psi\left(x_{1}, \ldots, x_{n}\right)$ there exists $\varphi$ such that for each database $D$ and elements $d_{1}, \ldots, d_{n}$ of the active domain of $D$,

$$
D \mid=\psi\left[d_{1}, \ldots, d_{n}\right] \quad \Longleftrightarrow \quad D \neq \text { adom } \varphi\left[d_{1}, \ldots, d_{n}\right] .
$$

We now show that this theorem has a simple proof assuming Proposition 1. Let $\sigma$ be the vocabulary having as sort symbols 1 and 2, and as relation symbols those of $L$, with $\sigma(R)=(1, \ldots, 1), n$ times, where $n$ is the arity of $R$ in $L$. We can view each database $D$ as a structure over $\sigma$, where $1^{D}$ equals the active domain of $D$ and $2^{D}$ equals its complement (taken in $U$ ). Given $\psi$, we construct a one-sorted formula $\bar{\psi}$ over $\bar{\sigma}$ inductively as follows:

- If $\psi$ is atomic then $\bar{\psi}:=\psi$;
- If $\psi$ is $(\neg \chi)$ or $(\chi \vee \theta)$ then $\bar{\psi}$ is $(\neg \bar{\chi})$ or $(\bar{\chi} \vee \bar{\theta})$ respectively;
- If $\psi$ is $(\exists x) \chi$ then $\bar{\psi}$ is $(\exists x)(1(x) \wedge \chi) \vee(\exists x)(2(x) \wedge \chi)$.

Clearly, $D=\psi\left[d_{1}, \ldots, d_{n}\right]$ iff $\bar{D}=\bar{\psi}\left[d_{1}, \ldots, d_{n}\right]$. By Proposition 1, there exists a formula $\phi$ over $\sigma$ such that $\bar{D} \models \bar{\psi}\left[d_{1}, \ldots, d_{n}\right]$ iff $D \models \phi\left[d_{1}, \ldots, d_{n}\right]$. By Lemma 2 (stated and proven below), we can write $\phi$ as a Boolean combination of formulas involving either only sort 1 or only sort 2 . Since $\phi$ has only free variables of sort 1 , every subformula involving sort 2 only can be evaluated independently of $D$ : it is a sentence in the language of pure equality saying something about the complement of the active domain, which is isomorphic to $U$. We can therefore replace each subformula of $\phi$ involving only sort 2 by its

[^1]absolute truth value on $U$, and obtain an equivalent formula $\varphi$ mentioning only variables of sort 1. These variables range over the active domain only, so the theorem is proven.

Lemma 2 Let $\sigma$ be a vocabulary with the property that the sort symbols in $\sigma$ can be partitioned into two parts, $X$ and $Y$, such that the sort of every relation name of $\sigma$ contains either only symbols from $X$, or from $Y$. Then every formula $\varphi$ over $\sigma$ is equivalent to a Boolean combination of formulas over $\sigma$ which mention either only variables of sorts in $X$ or of sorts in $Y$.

Proof. By induction on $\varphi$.

- If $\varphi$ is atomic and all variables occurring in it are either all of sorts in $X$ or of sorts in $Y, \varphi$ is already in the right form; otherwise, $\varphi$ is equivalent to false by the assumption on $\sigma$.
- The cases $\varphi$ is $(\neg \chi)$ or $(\chi \vee \theta)$ are trivial.
- If $\varphi$ is $(\exists x) \chi$, we know by induction that $\chi$ can be written as a Boolean combination of the right form. Rewrite $\chi$ in disjunctive normal form and distribute the quantifier over the terms of the disjunction. We are left with a disjunction of terms of the form $(\exists x)\left(\psi_{1} \wedge \cdots \wedge \psi_{l} \wedge \theta_{1} \wedge \cdots \wedge \theta_{m}\right)$, where each $\psi_{i}$ mentions only variables of sorts in $X$ and each $\theta_{i}$ mentions only variables of sorts in $Y$. If $x$ has sort in $X$, we rewrite the term as $(\exists x)\left(\psi_{1} \wedge \cdots \wedge \psi_{l}\right) \wedge \theta_{1} \wedge \cdots \wedge \theta_{m}$; if $x$ has sort in $Y$, we rewrite as $\psi_{1} \wedge \cdots \wedge \psi_{l} \wedge(\exists x)\left(\theta_{1} \wedge \cdots \wedge \theta_{m}\right)$. Now $\varphi$ is back in the right form.


[^0]:    ${ }^{1}$ The trivial formula defining the full relation of sort $\left(s_{1}, \ldots, s_{n}\right)$ is $\left(x_{1}=x_{1}\right) \wedge \cdots \wedge\left(x_{n}=\right.$ $x_{n}$ ), where $x_{i}$ is a variable of sort $s_{i}$ for $i=1, \ldots, n$.

[^1]:    ${ }^{2}$ R. Hull and J. Su. Domain independence and the relational calculus. Acta Informatica 31, 513-524 (1994).

